関係データベース入門

安藤 和敏
静岡大学工学部

平成 15 年 12 月 23 日

概 要

厳密に関係データベースを定義するとかなり長い文章になってしまいそうなので, 厳密さをある程度犠牲にして, 情報処理技術者試験 (基本情報) の関係データベースの正規化の問題が解ける程度の知識を簡潔に記述した.

1. 関係データベースとは何か?

関係データベースとは, 図 1.1 に示されるようないくつかの関連付けられた**テーブル** (table, 表) の集まりである. このデータベースは, 「科目」, 「学生」, 「履修」の 3 つのテーブルから構成されている. テーブルのより詳しい定義は第 2 節と第 3 節で述べる. それまでは, MS Excel で作られる表のようなものという理解で良い. これらのテーブルがどのように関連付けられるかということは, 第 5 節で述べる.

科目

科目名	単位数
データベース	2
人工知能	1

学生

学籍番号	氏名	住所
S1	小泉純一郎	東京
S2	加藤鉱一	秋田
S3	山崎拓	東京

履修

科目名	学籍番号	得点
データベース	S1	80
データベース	S2	100
人工知能	S2	50
人工知能	S3	70

図 1.1: 関係データベースの例

2. テーブル

テーブルは, 「構造」をもっている. 例えば図 1.1 のテーブル「履修」は, 科目名, 学籍番号, 得点という名前の付いた 3 つの列から成り, この 3 つ列がこのテーブルの構造を決定しているのであ

る. これらの列の名前を**属性** (attribute) と呼ぶ.

テーブルの構造の表現の仕方には何種類かある. テーブル「履修」は,

履修	科目名	学籍番号	得点

とか,

履修 (科目名, 学籍番号, 得点)

とか,

履修=科目名 + 学籍番号 + 得点

というように. 最近の情報処理技術者試験の出題を見た限りでは最初の表現が多くみられる.

テーブルは, いくつかの**レコード** (行) からなる. レコードは, そのテーブルの構造の属性に対応する**属性値**の並び (ベクトル) である. 例えば履修テーブルの最初のレコードは,

(データベース, S1, 80)

である.

3. 第 1 正規形

テーブルの形式はどんなものでも良いかというとそうではない. 関係データベースのテーブルは**第 1 正規形** (first normal form, 1NF) でなければならない.

第 1 正規形

テーブルの各属性の取る値は単一の属性値でなければならない. 簡単にいうと, 各属性の取る値は集合であったり属性値の並びであってはならない.

例えば, 表 3.1 の 1 番目のレコードは, (安藤和敏, { 読書, 音楽鑑賞, ドライブ, 映画鑑賞 }) という

氏名	趣味
安藤和敏	読書 音楽鑑賞 ドライブ 映画鑑賞
新妻清三郎	ハイキング 読書
関谷和之	サッカー
八巻直一	プラモデル ギター

表 3.1: 第 1 正規形でない例

ように, 属性「趣味」の値が集合になっている. よって, このテーブルは第 1 正規形ではない. 表 3.1 の構造は以下のように表される.

(氏名, { 趣味 })

ここで, { } は繰返しがあるということを意味しているのである. 集合を表しているわけではないので注意せよ. この表 3.1 を第 1 正規形にしたものが, 表 3.2 である.

氏名	趣味
安藤和敏	読書
安藤和敏	音楽鑑賞
安藤和敏	ドライブ
安藤和敏	水泳
安藤和敏	映画鑑賞
新妻清三郎	ハイキング
新妻清三郎	読書
関谷和之	サッカー
八巻直一	プラモデル
八巻直一	ギター

表 3.2: 表 3.1 を第 1 正規形にした表

4. 候補キーと主キー

図 1.1 のテーブル「学生」を図 4.1 に再掲した. このテーブルの各レコードは, 学籍番号という

学生

学籍番号	氏名	住所
S1	小泉純一郎	東京
S2	加藤鉱一	秋田
S3	山崎拓	東京

履修

科目名	学籍番号	得点
データベース	S1	80
データベース	S2	100
人工知能	S2	50
人工知能	S3	70

図 4.1: テーブル「学生」とテーブル「履修」

属性 (列) だけによって完全に指定できるということがわかるであろう. 例えば, 「山崎拓」のレコードを指定するためには, 「学籍番号が S3」のレコードと指定すれば良い. このような性質を持った属性がまだあるだろうか? 氏名という属性がそうだと思うかも知れないけれど, もし同性同名の学生がいたらそのような性質は成り立たないであろう.

次に, テーブル「履修」を見てみよう. このテーブルにおいてもテーブル「学生」のように一つの属性だけによって, レコード (行) を完全に指定できるであろうか? 答えば No である. しかし, 属性の組 (科目名, 学籍番号) を指定すれば, 各レコードが完全に指定できることがわかるであろう.

以上の例で見てきたように, あるテーブルの属性の組は,

性質 1. そのテーブルのレコードをその組だけで完全に指定できる. かつ,

性質 2. その属性たちのどの属性を除いても性質 1 を満たさない.

ようなものは, **候補キー** (candidate key) と呼ばれる.

例えばテーブル「履修」において, (科目名, 学籍番号) の組は候補キーである. また, テーブル「学生」においては (学籍番号) が候補キーである. (テーブル「学生」において, (学籍番号, 氏名) の組は性質 1 を満たすが, 性質 2 を満足したにので候補キーではない.) 一般には候補キーは, 一意には定まらない. 表 4.1 に示したテーブル「社員」には, (社員番号) も (健康保険番号) も候補キーである. いくつかある候補キーの中から, データベースの設計者が勝手に一つ選んだ候補キーは**主キー** (primary key) と呼ばれる. 候補キーが一つしかない場合には, もちろん候補キー=主キーである. (基本情報試験問題では, 候補キーは一つしかない場合が多いので, 候補キーと主キーの違いをあまり意識しなくても良いのかも知れない.) 主キーは, 属性名に下線を付けることによって表される.

社員

社員番号	社員名	給与	所属	健康保険番号
0650	阿部昭博	50	K55	80596
1508	斎藤美枝子	40	K41	81403
0231	神田茂	60	K41	80201
2034	渡辺和代	30	K55	81998

表 4.1: テーブル「社員」

例えば, 図 1.1 のデータベースの構造を主キーも含めて表現すると図 4.2 のようになる.

科目	科目名	単位数

学生	学籍番号	氏名	住所

履修	科目名	学籍番号	得点

図 4.2: 図 1.1 の構造

5. 外部キーと 1 対多の関係

これらの 3 つのテーブルは, ばらばらに存在するわけではない. テーブル「履修」の属性「科目名」は, テーブル「科目」における主キーとなっている. このように, あるテーブル T の属性で他のテーブル T' における主キーとなっているものを T の**外部キー** (foreign key) と呼ぶ. テーブル「履修」の属性「学籍番号」はテーブル「学生」の主キーであるから, テーブル「履修」の属性「学籍番号」も外部キーである.

テーブル「科目」の属性「科目名」は, このテーブルの主キーであるからこのテーブルにおいては, 同じ科目名を持つレコードは 1 つしかない (図 5.1 を見よ). しかし, テーブル履修においては, 同じ科目名を持つレコードは複数存在する. このような外部キーを介したテーブル間の関係は, **1 対多**の関係と呼ばれる.

図 5.1: 1 対多の関係

6. 関数従属性

表 6.1 にあるテーブル「注文」を見てみよう. ここでは, どの顧客も同一の商品を重複して注文しないと仮定する. この表が満すべき性質として, 以下のものが挙げられることが理解できるであ

注文

顧客名	商品名	数量	単価	金額
A 商店	テレビ	3	198,000	594,000
B マート	テレビ	10	198,000	1,980,000
B マート	洗濯機	5	59,800	299,000
C 社	餅つき機	1	29,800	29,800

表 6.1: テーブル「注文」

ろう.

- 商品名が同じならば, 単価も同じである.
- 数量と単価が等しいならば, 金額も等しい.
- 商品名と数量が等しいならば, 金額も等しい.
- 金額と数量が等しいならば, 単価が等しい.

このように, テーブルの構造において

「もし属性 A_1〜属性 A_k が等しいならば, 属性 B も等しい」

というような性質を**関数従属性** (functional dependency) と呼び,

$$\{A_1, \cdots, A_k\} \to B$$

と表す. したがって, 表 6.1 における関数従属性を記号で表せば,

{ 商品名 } → 単価, { 数量, 単価 } → 金額, { 商品名, 数量 } → 金額, { 金額, 数量 } → 単価

となる. この他にも, 自明な従属性が成り立つ. また, (顧客名, 商品名) は主キーであるから,

{ 顧客名, 商品名 } → 数量, { 顧客名, 商品名 } → 単価, { 顧客名, 商品名 } → 金額

はもちろん成り立つ.

X を属性からなる集合とすると任意の $a \in X$ に対して $X \to a$ である. これを, **自明な従属性**と呼ぶ.

関数従属性は, 図 6.1 のように矢線を用いて図示すると見通しが良くなる. 図が繁雑になるので全ての従属性を書きこんではいないことに注意せよ.

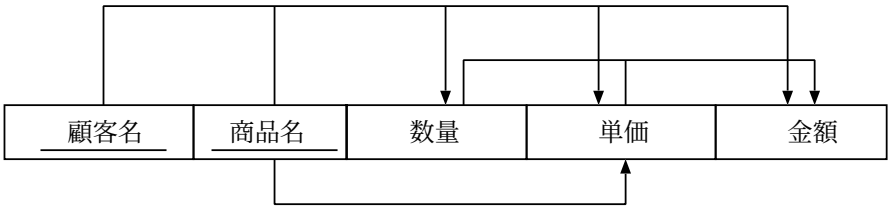

図 6.1: 関数従属性の図示

7. 第 2 正規形

X を属性から成る集合として, a を一つの属性とする. 関数従属性 $X \to a$ が成り立っていて, X のどんな真部分集合 $X'(\subset X)$ に対しても $X' \to a$ が成り立たないとき, a は, X に**完全従属** (fully dependent) しているという.

第 2 正規形

表は,

1. 第 1 正規形であり, かつ,
2. 各非キー属性は, 各候補キーに完全従属している

とき, **第 2 正規形** (second normal form, 2NF) と呼ばれる.

基本情報の試験問題に見られるように, 主キーが唯一の候補キーであるような場合では,

1. 第 1 正規形であり, かつ,
2. 主キーに含まれていない各属性は, 主キーに完全従属している

が第 2 正規形である条件となる.

例えば, 表 6.1 のテーブル「注文」はもちろん第 1 正規形である. また, 主キーである (顧客名, 商品名) の組がこのテーブルの唯一の候補キーである. 主キーに含まれていない属性「単価」を考えると,

$$\{ \text{商品名} \} \to \text{単価}$$

が成り立っているが, この関数従属性の左辺の { 商品名 } は主キー { 顧客名, 商品名 } の真部分集合である. つまり, 属性「単価」は, 主キーに完全従属していない. したがって, このテーブル「注文」は第 2 正規形では**ない**.

注文

顧客名	商品名	数量	金額
A 商店	テレビ	3	594,000
B マート	テレビ	10	1,980,000
B マート	洗濯機	5	299,000
C 社	餅つき機	1	29,800

商品

商品名	単価
テレビ	198,000
洗濯機	59,800
餅つき機	29,800

図 7.1: テーブル「注文」の分解

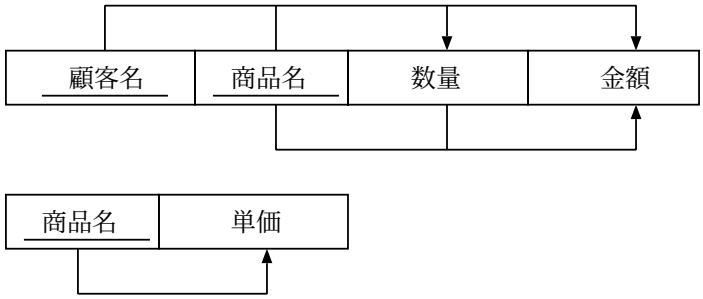

図 7.2: 第 2 正規形になった注文データベース

第 2 正規形にするためには, 主キーに完全従属していない属性を切り離して別の表を作ればよい (図 7.1 を見よ). 分解して得られた 2 つのテーブルは, どちらも第 2 正規形であるというこは, 容易に確かめられるから各自確かめてみよ. 図示すると図 7.2 のようになる.

8. 第 3 正規形

表 8.1 で示されるテーブル「社員」を考えよう. 関係従属性を図示すると, 図 8.1 のようになる.

社員

社員番号	社員名	給与	所属	勤務地
0650	阿部昭博	50	K55	神奈川
1508	斎藤美枝子	40	K41	東京
0231	神田茂	60	K41	東京
2034	渡辺和代	30	K55	神奈川
2100	鶴窪孝博	40	K58	静岡

表 8.1: テーブル「社員」

この図から明らかなように, テーブル「社員」は第 2 正規形である.

ここで, { 社員番号 } → 勤務地 という従属性は,

{ 社員番号 } → 所属 と { 所属 } → 勤務地

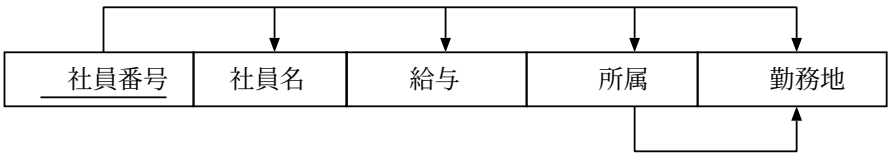

図 8.1: テーブル「社員」の関係従属性

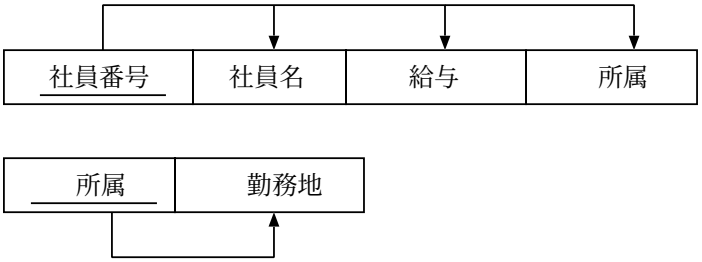

図 8.2: テーブル「社員」の正規化

という 2 の従属性が推移的に作用して生じたものであるとみなせる. 厳密な定義は省略するが, このとき,

属性「勤務地」は, 社員番号に**推移的に従属する**

という.

第 3 正規形

テーブルは,

1. 第 2 正規形であり, かつ,
2. 各非キー属性は, 各候補キーに推移的に従属しない

とき, **第 3 正規形** (third normal form, 3NF) であるという.

第 3 正規形でないテーブルは, 第 2 正規化のときと同じように, 推移的な従属性がなくなるように, テーブルを分解してやればよい (図 8.2 を見よ).

9. 過去問研究

9.1. 平成 12 年秋・午前問 51

表 9.1 を正規化しろという問題である. この表の構造は以下のように表される.

受注表 (受注 No, 受注日, 受注先, { 商品, 数量, 単価 }, 合計金額)

これを第 1 正規形にすると,

(受注 No, 受注日, 受注先, 商品, 数量, 単価, 合計金額)

受注表

受注 No	受注日	受注先	商品	数量	単価	合計金額
1	00/10/01	A	S T S	3 2 1	1,000 950 1,000	4900
2	00/10/01	B	U V	10 5	1,200 1,800	22,000
3	00/10/02	B	T	8	950	7,600
4	00/10/02	C	U	25	1,200	30,000
⋮	⋮	⋮	⋮	⋮	⋮	⋮

表 9.1: 第 1 正規形でない表

になる. 第 2, 第 3 正規形になるように分解するために, このテーブルの関数従属性を調べてみると, 図 9.1 のようになる. この図から, (受注 No, 商品, 数量) が唯一の候補キーであることがわかるので, これを主キーとする. (この場合は, 矢印が 1 本も入ってこない属性たちを集めて主キーとした.)

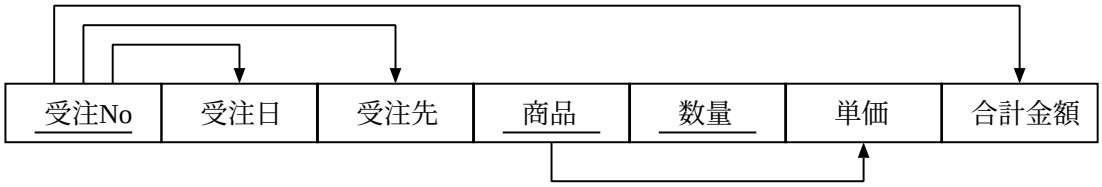

図 9.1: 関数従属性

このままでは, まだ第 2 正規形ではない. 非キー属性である単価は主キーの真部分集合である { 商品 } に従属している. また, 同じく非キー属性である受注日, 受注先, 合計金額は, 主キーの真部分集合である { 受注 No} に従属している. したがって, 3 つのテーブルに分解して図 9.2 のようにすれば, 第 2 正規形である. この時点で, 推移的な従属性も存在しないから, このテーブルたちは第 3 正規形でもある.

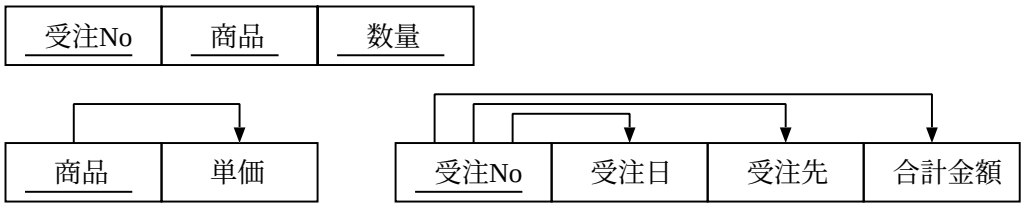

図 9.2: 正規化されたテーブルたち

9.2. 平成 14 年春季・午前問 67

以下のような構造を持った表を第 3 正規形にしろという問題である.

技能記録 (従業員番号, 従業員氏名, { 技能コード, 技能名, 技能経験年数 })

とりあえず, 第 1 正規形にして, 関数従属性を調べてみる. さきほどと同様に矢印が入ってこない

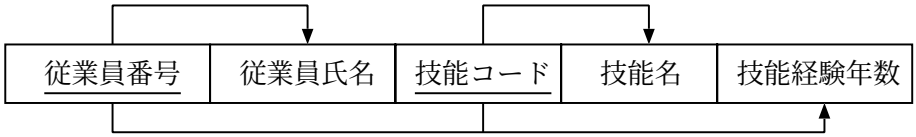

図 9.3: 第 1 正規形の技能記録

属性を集めて主キーとした. このテーブルでは 2 箇所で, 第 2 正規形の条件が破られている. 即ち, { 従業員番号 } → 従業員氏名 と { 技能コード } → 技能名 である. この 2 つに対してテーブルを分解して, 図 9.4 のように 3 つのテーブルにする. これで, 第 2 正規形になったのであるが, 既に第 3 正規形にもなっている.

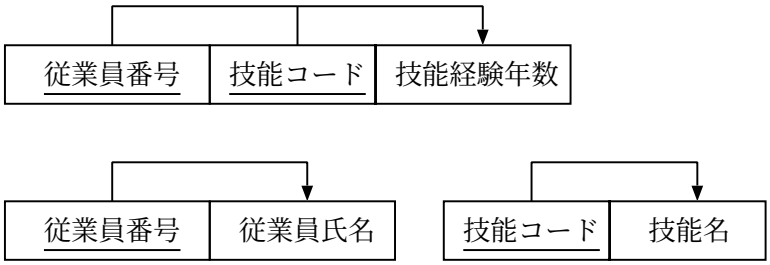

図 9.4: 第 3 正規形の技能記録

参考文献

[1] 増永良文: リレーショナル・データベース入門. サイエンス社, 1991 年.

[2] 矢沢久雄: コンピュータはなぜ動くのか. 日経 BP 社, 2003 年.